Devoir Maison cryptographie

Afouchal Ayoub , Elhabar Moussa

April 2020

1 Excercie 1

1. Ouvrir le fichier Main.c dans le dossier "EXERCICE1" et un terminal et tapez la commande "make" , pour changer la valeur de la fonction de filtrage ou les clées des différents LFSR ouvrez le fichier Makefile.

2. On calcule théoriquement la correlation entre la sortie du générateur S_i et la sortie du LFSR par la relation suivant :

   On note Nb le nombre de fois que la sortie du LFSR est égale à la sortie de la fonction de filtrage .

   $$\frac{\texttt{Nb}}{\texttt{Nombre de possibilitées}}$$

x_0	x_1	x_2	$F(x_0, x_1, x_2)$
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

   Donc on a :

   - x_0 : $\frac{2}{8} = \frac{1}{4} = 25\%$
   - x_1 : $\frac{2}{8} = \frac{1}{4} = 25\%$
   - x_2 : $\frac{6}{8} = \frac{3}{4} = 75\%$

3. Attaque par corrélations :
   Cette attaque consiste en l'exploitation des correlations qui existent entre la sortie du générateur et les lfsr , pour attaquer le générateur on teste toutes les initialisation possible de chaque lfsr (2^{16}).
   à chaque initialisation on génère une suite chiffrante de la même taille que le chiffrré connu puis on clalcul la corrélation entre cette suite chiffrrante et le généré et on compare le résultat à la corrélation que l'on connait (voir le fichier attack-diviser.c).

4. L'éstimation du nombre de bits de la suite chiffrante que l'attaquant doit connaitre pour mettre en oeuvre son attaque est de 16 bits.

   - La compléxité en temps de cette attaque est $2^{16} * 3$.
   - la compléxite en mémoire de cette attaque est de 2^{48}.

5. Voir le fichier AttackDiviser.c pour exécuter le programme tapez "make attack" .

6. Pour rendre l'attaque contre ce générateur la plus difficile possible , il faut trouver une fonction F qui a une correlation de 50% avec les 3 LFSR.
   On peut prendre comme exemple de fonction F , la fonction : 01100110 .

2 Exercice 2 :

1. On a : $(x_0^L, x_0^R) =$ (`0x45019824; 0x51023321`).

   Et : $k_0=$ `0x01020304` , $k_1=$`0x98765432.`

   - `0x45019824 = 01000101000000011001100000100100.`
   - `0x51023321 = 01010001000000100011001100100001.`

     Pour x_1^L :

   - `0x51023321` $\oplus$ `0x45019824` = `00010100000000111010101100000101.`
   - `00010100000000111010101100000101 <<7` = `00000001110101011000001010001010.`
   - $k_0 =$ `0x01020304` = `00000001000000100000001100000100.`
   - $x_1^L = k_0 \oplus$`00000001110101011000001010001010`=`00000000110101111000000110001110` = `d7818e.`

Pour x_1^R :

- $x_1^L \oplus$ `0x51023321` = `01010001110101011011001010101111.`
- `01010001110101011011001010101111<<7` = `11101010110110010101011110101000.`
- x_1^R =`0x98765432`$\oplus$`11101010110110010101011110101000`
  =`01110010101011110000001110011010`= `72AF039A.`

2. - Le chiffrement sous forme de système d'équations est :

$$\begin{cases} x_1^L = k_0 \oplus F(x_0^L \oplus x_0^R) \\ x_1^R = k_1 \oplus F(x_0^R \oplus x_1^L) \end{cases}$$

On note F une fonction de rotation de 7 bits.

On a :

$$\begin{cases} k_0 = x_1^L \oplus F(x_0^L \oplus x_0^R) \\ k_1 = x_1^R \oplus F(x_0^R \oplus x_1^L) \end{cases}$$

- Pour résoudre le système , il faut seulement avoir le texte clair et le texte chiffré comme ca on peut trouver la clé .
Voir le programme "main.c" dans le dossier EXERCICE2 tapez la commande "make" pour exécuter le programme.

3. Généralisation de la cryptanalyse au chiffrement complet (12 tours) :
On a :

$$\begin{cases} k_0 = x_{i+1}^L \oplus F(x_i^L \oplus x_i^R) \\ k_1 = x_{i+1}^R \oplus F(x_i^R \oplus x_{i+1}^L) \end{cases}$$

On suppose qu'on a 2 deux couples : clair , chiffré (a_0, a_{12}) et (b_0, b_{12})
On suppose que : $a_0 = b_1$ et $a_{11} = b_{12}$
Donc : $a_i = b_{i+1}$.
alors : $k_0 = a_{12}^L \oplus F(a_{11}^L \oplus a_{11}^R)$
donc on remplace dans la formule .
On aura : $k_0 = a_{12}^L \oplus F(b_{12}^L \oplus b_{12}^R)$
Donc la on peut calculer k_0.
avec le k on peut calculer le a_0 et le comparer avec le clair de l'entrée si c'est bon donc c'est la bonne clé . On fait la meme chose pour k_1.

4. Tapez la commande "make attack" pour éxecuter le progamme

5. Non , ajouter plus de tours rendra jamais le chiffrement plus solide parceque la fonction de rotation F utilise la meme clé pour chaque tout du coup on peut appliquer l'attaque a chaque fois parceque a chaque fois on peut deduire la clé secréte.

6. Pour améliorer ce chiffrement , il faut changer les clées k_0 et k_1 à chaque tour .